Guidelines for Creating Content for the PSP™ (PlayStation® Portable) Internet Browser

Version 5.00

© 2008 Sony Computer Entertainment Inc.
All Rights Reserved.

[Copyright and Trademarks]
"UMD" and "PlayStation" are registered trademarks of Sony Computer Entertainment Inc.
"PSP" is a trademark of Sony Computer Entertainment Inc.
"Memory Stick" is a trademark of Sony Corporation.
"XMB" is a trademark of Sony Corporation and Sony Computer Entertainment Inc.
This document contains some copyright materials of ACCESS CO.,LTD.
PSP™ contains NetFront Internet browser software of ACCESS CO.,LTD.
Copyright© 1996-2008 ACCESS CO.,LTD.
ACCESS and NetFront are trademarks or registered trademarks of ACCESS CO.,LTD. in Japan and other countries.
Java and JavaScript are trademarks or registered trademarks of Sun Microsystems,Inc. in the United States and other countries.
Flash, Macromedia and Macromedia Flash are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.
VeriSign is a trademark or registered trademark of VeriSign,Inc. in the United States and other countries.
RSA is a trademark or registered trademark of RSA Security Inc. in the United States and other countries.
"Mozilla" is a registered trademark of the Mozilla Foundation.
All other trademarks are the properties of their respective owners.

[Terms and Conditions]
All rights (including, but not limited to, copyright) pertaining to all the materials and information that are made available from this Guideline are managed, owned, or used with permission by, SCEI. All materials in this Guideline are protected by copyright laws, and other applicable laws. Except for personal, non-commercial, internal use, you are prohibited from using (including, without limitation, copying, modifying, reproducing in whole or in part, uploading, transmitting, distributing, licensing, selling and publishing) any of the materials, without obtaining SCEI's prior written permission.
The contents of this Guideline are not guaranteed to be valid for future versions of system software.
SCEI OR ANY OF ITS AFFILIATES SHALL NOT BE LIABLE FOR ANY DAMAGES ARISING OUT OF ANY CHANGE OF THE SYSTEM SOFTWARE SPECIFICATIONS.

Table of Contents

1 Introduction 5

Related Documents 5

2 Basic Browser Functionality 6

HTTP 6

Cookies 6

Cache 7

HTTP Authentication 7

Proxy Authentication 8

Schemes 8

HTML 8

SSL/TLS 9

CSS 10

DOM 10

JavaScript 10

Screen Display 10

Character Display 10

Image Display 13

Screen Displays 14

Plug-ins 15

3 Browser Application Functionality 16

Addresses 16

Titles 16

Character Input 17

Bookmarks 18

Tabs 18

Downloads 19

Uploads 22

Automatic HTTP Authentication 22

4 Client Identification Information 24

Request Header 24

JavaScript 27

5 Flash® Player Plug-in 29

Version 29

Supported Devices 29
Unsupported Functions 29
Known problems and restrictions 30
6 PSP™ Plug-in 32
Supported Functions 32
Plug-in Definition 32
Writing a Return Value to the Application 32
Shutting Down the Internet Browser 33
Sample 34
7 PSP™ Extended Plug-in 36
Supported Functions 36
Plug-in Definition 36
Getting Internet Browser Extended Header Information 36
Getting the Maximum Content Heap Size Used 37
Getting the Total Content Heap Size 38
Sample 38
8 Appendix 41
Detailed Specifications for HTML 41
Detailed Specifications for CSS 43
Detailed Specifications for JavaScript 44
Detailed Specifications for DOM 44
Initial Values and Constraints 50
9 Document History 52

1 Introduction

This document describes required client specifications and other information as well as guidelines to be followed when creating Web content for the PSP™ Internet browser.
It should be referenced when creating such content. However, the Internet browser client specifications may also be changed to an extent considered appropriate mainly for fixing bugs or improving quality.
The contents of this document are not guaranteed to be valid for future versions of system software. In this document, where differences exist depending on the version of system software, these are clearly indicated using notation such as "2.00" or "5.00".

Related Documents

For details about standards that the Internet browser conforms to or supports, refer to the following documents.

Protocol

[RFC2616] Hypertext Transfer Protocol – HTTP/1.1

PERSISTENT CLIENT STATE HTTP COOKIES

http://wp.netscape.com/newsref/std/cookie_spec.html

Markup Language

HTML 4.01 Specification http://www.w3.org/TR/html401/

XHTML™ 1.1 –Module-Based XHTML http://www.w3.org/TR/xhtml11/

XHTML™ Basic http://www.w3.org/TR/xhtml-basic/

CSS

Cascading Style Sheets, level1 http://www.w3.org/TR/REC-CSS1

Cascading Style Sheets, level2 http://www.w3.org/TR/REC-CSS2/

JavaScript

Standard ECMA-262

http://www.ecma-international.org/publications/standards/Ecma-262.htm

2 Basic Browser Functionality

The PSP™ Internet browser conforms to Internet standard protocols and supports standard markup languages such as HTML 4.01 and XHTML 1.1 as well as CSS, DOM, and JavaScript. This chapter describes this basic functionality.

HTTP

(1) Supported protocols

The Internet browser supports the following standard protocols.

- HTTP/1.0
- HTTP/1.1

(2) Supported methods

The Internet browser supports (uses) the following HTTP methods.

- GET
- POST
- HEAD

(3) Supported functions

The Internet browser supports the following functions.

- Redirect
- Keep-Alive
- Cookie
- Cache
- HTTP Authentication

Cookies

Cookies are not always saved. Cookie saving is controlled by a user setting and a specification of the application that called the Internet browser. Cookie saving may also be canceled by a user confirmation.

(1) Supported attributes

The Internet browser supports the following cookie attributes.

- Expires
- Path
- Domain
- Secure

(2) Restrictions

The cookie function has the following restrictions.

Number of cookies saved

At most 20 cookies are saved per domain.

Cookie size

The maximum size of one cookie is 4K bytes.

Cache

(1) Cache size

The cache size is determined by a user setting and a specification of the application that called the Internet browser. The standard size is 512K bytes.

The cache may also be set or specified to be off.

(2) Restrictions

The cache function has the following restrictions.

Cache retention period

The cache is enabled only while the Internet browser is executing. When the Internet browser is terminated, the contents of the cache are erased.

The cache is also erased by a "delete cache" user operation. This operation can be performed even while the Internet browser is executing.

Disabling the cache

The use of a cache disable specification in a META tag in a page is not supported.

To disable the cache, the following HTTP response header must be specified.

Cache-Control: no-cache

HTTP Authentication

(1) Supported authentication methods

The Internet browser supports the following HTTP authentication methods.

- Basic authentication
- Digest authentication

(2) Input information retention

The user name and password that were entered for basic authentication or digest authentication are saved for each URL and automatically entered during the next authentication.

This input information, which is saved for the PSP™ console, is valid until the user

performs a "delete authentication information" operation or a "set up initialization" operation is performed.

(3) Automatic HTTP Authentication

The Internet browser provides a function that uses authentication information specified from the application to automatically perform HTTP authentication without making the user enter a user name and password.

For details, see the section entitled "Browser Application Functions."

Proxy Authentication

(1) Supported authentication methods

The Internet browser supports the following proxy authentication methods.

- Basic authentication
- Digest authentication

(2) Automatic proxy authentication

The Internet browser provides a function that uses authentication information specified by the user to automatically perform proxy authentication without having to display an input dialog for the user name and password.

(3) Supporting systems

3.00 or later

Schemes

The Internet browser supports the following schemes.

- http
- https

Although the browsing of content on a Memory Stick™ using a file scheme is also functionally supported, that operation and its future support are not guaranteed.

HTML

The Internet browser supports the following standard markup language specifications.

- HTML 4.01
- XHTML 1.1
- XHTML Basic

For support details, see "Appendix: Detailed Specifications for HTML."

SSL/TLS

The Internet browser conforms to the following standard.

- SSL v3.0

(1) Root certificates

The following root certificates are built into the PSP™ and used when the Internet browser makes an SSL connection.

Certificate	In System
Verisign Root CA	2.00 or later
Verisign Class 1 Public Primary CA	2.00 or later
Verisign Class 2 Public Primary CA	2.00 or later
Verisign Class 3 Public Primary CA	2.00 or later
Verisign Class 1 Public Primary CA G2	2.00 or later
Verisign Class 2 Public Primary CA G2	2.00 or later
Verisign Class 3 Public Primary CA G2	2.00 or later
Verisign Class 4 Public Primary CA G2	2.00 or later
Verisign Class 1 Public Primary CA G3	2.00 or later
Verisign Class 2 Public Primary CA G3	2.00 or later
Verisign Class 3 Public Primary CA G3	2.00 or later
Verisign Class 4 Public Primary CA G3	2.00 or later
Verisign RSA Secure Server CA	2.00 or later
Verisign Time Stamping Authority CA	2.00 or later
RSA Root CA	2.00 or later
RSA Security Root CA 1024 (Valicert Class 3 CA)	2.00 or later
RSA Security Root CA 2048 V3	2.00 or later
GeoTrust Root CA	2.00 or later
GeoTrust Global CA	2.00 or later
GeoTrust Equifax Secure CA	2.00 or later
GeoTrust Equifax Secure eBusiness CA-1	2.00 or later
EnTrust Root CA	2.00 or later
EnTrust.net Secure Server CA (CPS)	2.00 or later
Valicert Root CA	2.50 or later
Valicert Class 2 CA	2.50 or later
OmniRoot (CyberTrust CA)	2.50 or later
Omni Baltimore CyberTrust CA	2.50 or later
Omni GTE CyberTrust Global Root CA	2.50 or later
Omni GTE CyberTrust Root CA	2.50 or later
Omni Globalsign Root CA	2.50 or later
Thawte Root CA	2.70 or later
Thawte PremiumServer CA	2.70 or later
Thawte Server CA	2.70 or later
COMODO Root CA	5.00 or later
AddTrust External CA Root	5.00 or later
AAA Certificate Services	5.00 or later
UTN-USERFirst-Hardware	5.00 or later

CSS

The Internet browser supports the following standards.

- CSS1
- Part of CSS2

For support details, see "Appendix: Detailed Specifications for CSS."

DOM

The Internet browser supports the following standards.

- DOM level 1
- Part of DOM level 2

For support details, see "Appendix: Detailed Specifications for DOM."

JavaScript

The Internet browser supports the following standard.

- Part of JavaScript 1.5

For support details, see "Appendix: Detailed Specifications for JavaScript."

Screen Display

The Internet browser uses the entire PSP™ display screen to display pages. The screen size, which is 480x272, is not reduced or enlarged.
Although a page that exceeds the screen size can be scrolled by the user, no scroll bars are displayed.
The operation menu is displayed superimposed on the page.
Otherwise, the page display size cannot be dynamically changed by the user.

Character Display

(1) Font size

The Internet browser supports five font sizes. More sizes may be specified in the content, or the point size may be directly specified. The user can select one of three character sizes, either "large," "standard," or "small".
The Internet browser maps the size specification from the content to the five font sizes according to the following rules.

Point specification

The point specification in the content is mapped by multiplying the points by 1 when the character size is "large," by 3/4 when the character size is "standard," and by 1/2 when the character size is "small."

Points	Font Size
16 pt or greater	Size 4 (maximum)
12 pt to 15 pt	Size 3
10 pt to 11 pt	Size 2
8 pt to 9 pt	Size 1
7 pt or less	Size 0 (minimum)

Size specification

The size specification is mapped as follows for each of the character sizes "large," "standard," and "small."

Size Specification	Font Size (Large)	Font Size (Standard)	Font Size (Small)
7	Size 4	Size 4	Size 3
xx-large	Size 4	Size 3	Size 2
6	Size 4	Size 3	Size 2
x-large	Size 3	Size 2	Size 0
5	Size 3	Size 2	Size 0
large	Size 3	Size 1	Size 0
4	Size 3	Size 1	Size 0
medium	Size 2	Size 1	Size 0
3	Size 2	Size 1	Size 0
small	Size 1	Size 0	Size 0
2	Size 1	Size 0	Size 0
x-small	Size 0	Size 0	Size 0
1	Size 0	Size 0	Size 0
xx-small	Size 0	Size 0	Size 0

(2) Font family

The Internet browser allows the display font to be changed by specifying the font family, but for Latin fonts only.

The following font families can be specified.

Font Families

- Roman
- Arial
- Serif
- Sans-Serif

(*) However, specifying the font family in a font tag is not permitted. The font family can only be specified using a style attribute in a span tag as shown below.

Example

```
<span style="font-family:Roman">Roman</span>
```

(3) Font style

The Internet browser supports italic and bold font styles and font weight specifications, but only for Latin fonts.

The following font styles can be specified.

Font Styles

- italic
- bold

Examples

```
<b>bold</b> and <i>italic</i>
<span style="font-style:italic">italic</span>
```

(4) Supported character sets

The Internet browser determines and displays the character set based on a specification in a META tag in the HTML content.

The following character sets are supported for display.

US-ASCII	ASCII
ISO-8859-1	Western Europe
Windows-1252	Western Europe
ISO-8859-2 *	Central Europe
Windows-1250 *	Central Europe
ISO-8859-3 *	Southern Europe
ISO-8859-4 *	Baltic languages
Windows-1257 *	Baltic languages
ISO-8859-5	Cyrillic characters
Windows-1251	Cyrillic characters
KOI8-R *	Cyrillic characters
ISO-8859-7 *	Greek
Windows-1253 *	Greek
ISO-8859-9	Turkish
Windows-1254 *	Turkish
GB2312	Simplified Chinese
GBK	Simplified Chinese
GB18030	Simplified Chinese
Big5	Traditional Chinese
EUC-KR	Korean
ISO-2022-JP	Japanese
Shift_JIS	Japanese
EUC-JP	Japanese
UTF-8	Unicode

Note that for the character sets marked with an asterisk (*), some characters cannot be displayed properly because of the relationships with the character types of the PSP™ built-in fonts.

(5) Limitations

The following limitations apply to character display.

Character set discrimination

The Internet browser displays characters in the character set that is specified in a META tag as follows.

<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />

If no character set is specified in a META tag, the operation is not guaranteed to be defined.

Whether

- a specific character set is selected according to the console's display language setting, or
- Unicode (UTF-8) is always selected

may change depending on the release.

Frame Handling

On a page that uses a frame or iframe, if the frame file does not contain a character set specification, then even if the base file contains a character set specification, there is no guarantee that it will be inherited.

Image Display

(1) Supported formats

The Internet browser supports the display of images in GIF, PNG, and JPEG formats. Support details for each format are as follows.

GIF	GIF87a, GIF89a Interlaced, noninterlaced, transparent color, animation
PNG	Interlaced, noninterlaced, transparent color, semitransparent (α channel) (*) Animation (MNG) is not supported
JPEG	Baseline DCT, progressive DCT

(2) Conserve Memory setting

With PSP™ system software version 3.10 or later, the amount of heap memory used for displaying images can be reduced by setting "Conserve Memory" under "View Settings" to "On."

In this case, the Internet browser displays images in content according to the following conditions.

Common	・Images are decoded using 16-bit color. ・If vertical x horizontal exceeds 130560 pixels, decoding is cut back so as not to exceed 130560 pixels.
PNG	・Semitransparency (α channel) is disabled.

(3) Limitations

The following limitations apply to image display.

Image size

To display an image, heap memory is temporarily used in an amount equal to the number of pixels in the image multiplied by 4 bytes.

Since the remaining amount of heap memory varies according to the heap memory size that is assigned by the application that called the Internet browser, the cache size setting, and how other tabs are being used, in some cases it may not be possible to display the image.

When an image cannot be displayed, a mark is displayed in white space equivalent to the image size or the size specified by the img tag.

Non-display setting

The user can specify that images and animated images are not to be displayed.

When an image is not displayed, a mark is displayed in white space equivalent to the image size or the size specified by the img tag.

When animated images only are not to be displayed, the first frame of the animation is displayed as a still image.

Conserve Memory

If an image cannot be decoded properly when the conserve memory setting is on, an is displayed instead of the image, in a space with the same size as the image or with the size specified in the img tag.

Screen Displays

The Internet browser supports screen display mode switching. Note that an unintended layout of the content may end up getting displayed depending on the display mode setting.

The following display modes are available.

- Standard

  The original page size and layout are displayed as is.
- Just fit mode

  The original layout is preserved as is and displayed to match the width of the screen. The character size may become smaller compared to Standard mode. Also, wrapping may automatically be performed.
- Smart fit mode

  The original layout is reconfigured and displayed to match the width of the screen. The character size may become smaller compared to Standard mode. Also, wrapping

may automatically be performed.

Plug-ins

The Internet browser supports plug-in functions.

(1) Supported plug-ins

The following plug-ins are supported.

Plug-in Name	MIME-Type	Supporting System
Flash® Player plug-in (*)	application/x-shockwave-flash	2.70 or later
PSP™ plug-in	application/x-psp-plugin	2.70 or later
PSP™ extended plug-in	application/x-psp-extplugin	2.80 or later

(*) Contains Macromedia® Flash® Player technology by Adobe. For details about each plug-in, see the corresponding section.

(2) Restrictions

The plug-in function has the following restrictions.

Startup Confirmation

A user confirmation dialog may be displayed when the plug-in is started up depending on the version of system software.

3 Browser Application Functionality

The PSP™ Internet browser has many application functions besides the basic browsing functions of tabs, bookmarks, and file downloading.
This chapter explains aspects of these functions that are related to content creation.

Addresses

The Internet browser has no specific constraints on the length of addresses that are used for links and image references in content. However, when handling address strings in the console such as when editing addresses or registering bookmarks, a restriction applies in which a single address can be no more than 512 characters in length.
Since an address that exceeds 512 characters is automatically cut to one that is less than or equal to 512 characters, be particularly careful with pages that may have been bookmarked.
The following functions are affected by this constraint.

- Editing the address of the page that is currently being displayed
- Registering a bookmark

Titles

The Internet browser has no specific constraints on the length of a content title. However, when handling title strings in the console such as when registering bookmarks, a restriction applies in which a single title can be no more than 512 characters in length.
Since a title that exceeds 512 characters is automatically cut to one that is less than or equal to 512 characters, be particularly careful with pages that may have been bookmarked.
The following function is affected by this constraint.

- Registering a bookmark

Also, when a long title is displayed, it will be shortened or automatically scrolled. To avoid this, the maximum number of characters in a title should be approximately 20 for Japanese, Korean and Chinese and 30 for other languages.
The following functions are affected by this constraint.

- Displaying the title bar (automatic scrolling)
- Displaying the list of personal bookmarks (shortened)
- Displaying the history list (shortened)

Character Input

(1) Supported functions

The Internet browser allows the user to input characters such as when entering addresses. Although characters are entered by using the on-screen keyboard, the maximum number of characters that can be entered is 512 or 256.

The following functions are affected by this constraint.

- Address entry
- Address entry for the home page setting
- Address entry for the proxy server setting
- User name entry for the proxy authentication setting (*) 256 characters
- Password entry for the proxy authentication setting (*) 256 characters
- Address entry for bookmark editing
- Title entry for bookmark editing
- Form entry
  - input (text, password, file)
  - textarea
  - forms within Flash® content
- File name entry for downloading, saving a link destination, or saving an image
- User name and password entry in HTTP authentication dialog (*) 256 characters
- User name and password entry in proxy authentication dialog (*) 256 characters
- Character entry in JavaScript prompt() dialog (*) 256 characters

The languages that can be entered also depend on on-screen keyboard support. The following languages can be entered according to the display language setting.

Supported Input Language	Supporting System
German	2.00 or later
English	2.00 or later
Spanish	2.00 or later
French	2.00 or later
Italian	2.00 or later
Dutch	2.00 or later
Portuguese	2.00 or later
Russian	2.00 or later
Japanese	2.00 or later
Korean	2.50 or later

(2) Restrictions

The character input function has the following restrictions.

maxlength specification for the input tag

The maximum number of input characters is controlled by the maxlength attribute of the input tag, which is a function of the number of bytes, as follows.

maxlength specification value >= (no. of multibyte characters x 3 + no. of single-byte characters)

Consequently, the content must make necessary adjustments if multibyte code input is expected.

- When multibyte code input is expected, specify the expected number of characters x 3 for maxlength.
- In this case, three times as many single-byte code characters can be input, so be sure that the content performs appropriate postprocessing.

Bookmarks

The Internet browser allows bookmarks to be added on a Memory Stick™. The bookmark function is outlined below.

Maximum number of bookmarks that can be registered	1000
Registration items	Address, title, last access date (*) For a page that has no title, the address is registered instead of the title.
Maximum number of address characters	512
Maximum number of title characters	512
Duplicate registrations	Allowed

Tabs

The Internet browser supports the tab function. The tab function is outlined below.

Maximum number of tabs	1 to 3 (*) Varies from 1 to 3 depending on a specification of the application that called the Internet browser. Always 3 when the Internet browser is started up from the "Internet Browser" icon of the XMB™. (*) If an attempt is made to open a page by using a separate tab, which would cause the maximum number of tabs to be exceeded, the operation may be canceled by a user selection.

Target specification	Allowed (*) The same tab is used to open the page by specifying the same target. (*) If "_blank" is specified for the target, the page is always opened in a new tab.
JavaScript verification	Yes (*) If an attempt is made to open a page with JavaScript by using a separate tab, the operation may be canceled by a user selection.

Downloads

The Internet browser supports the downloading of data that is linked on a page. When downloading, the Internet browser evaluates the MIME-Type (Content-Type) and extension and automatically determines the appropriate folder on the Memory Stick™.

(1) Evaluation procedure

The evaluation procedure is different between downloading via a link and when the "save link destination" operation is performed.

Each of these evaluation procedures is described below.

Determined by a link

(1) Read the content header to get the MIME-Type.

(2) Get the extension from the Content-Disposition header field(*) or the URL.

(3) Decide whether or not the content is displayable from the MIME-Type and extension.

(4) If the content is displayable, display it. Then exit.

(5) Determine the download destination folder from the MIME-Type and extension.

(*) The Internet browser supports the Content-Disposition header field in the System 2.60 or later.

Save link destination

(1) Get the extension from the URL.

(2) Determine the download destination folder from the extension.

(2) Content to be displayed

Displayable content is as follows.

MIME-Type

- text/html
- text/plain
- image/gif
- image/png
- image/jpeg
- each plug-in (only when valid)

Extension

- html, htm
- txt
- gif
- png
- jpg, jpeg

(3) Determination of download destination folder

The following table shows relationships among the MIME-Type, extension, and download destination folder.

MIME-Type	Extension	Folder	File Name	Can be Changed	Supporting System
-	jpg	/PSP/PHOTO	(*) 1	Yes	2.00~2.71
-	jpg	/PICTURE (*) 3	(*) 1	Yes	2.80 or later
-	jpeg	/PSP/PHOTO	(*) 1	Yes	2.00~2.71
-	jpeg	/PICTURE (*) 3	(*) 1	Yes	2.80 or later
-	gif	/PSP/PHOTO	(*) 1	Yes	2.00~2.71
-	gif	/PICTURE (*) 3	(*) 1	Yes	2.80 or later
-	png	/PSP/PHOTO	(*) 1	Yes	2.00~2.71
-	png	/PICTURE (*) 3	(*) 1	Yes	2.80 or later
-	bmp	/PSP/ PHOTO	(*) 1	Yes	2.00~2.71
-	bmp	/PICTURE (*) 3	(*) 1	Yes	2.80 or later
-	tif	/PSP/ PHOTO	(*) 1	Yes	2.00~2.71
-	tif	/PICTURE (*) 3	(*) 1	Yes	2.80 or later
-	tiff	/PSP/PHOTO	(*) 1	Yes	2.00~2.71
	tiff	/PICTURE (*) 3	(*) 1	Yes	2.80 or later
audio/*	-	/MUSIC (*) 4	(*) 1	Yes	2.80 or later
x-audio/*	-	/MUSIC (*) 4	(*) 1	Yes	2.80 or later
	mp3	/PSP/MUSIC	(*) 1	Yes	2.00~2.71
	mp3	/MUSIC (*) 4	(*) 1	Yes	2.80 or later
	mp4	/PSP/MUSIC	(*) 1	Yes	2.00~2.71
	wav	/PSP/MUSIC	(*) 1	Yes	2.00~2.71
	wav	/MUSIC (*) 4	(*) 1	Yes	2.80 or later
	wma	/PSP/MUSIC	(*) 1	Yes	2.60~2.71
	wma	/MUSIC (*) 4	(*) 1	Yes	2.80 or later
	aa3	/PSP/MUSIC	(*) 1	Yes	2.60~2.71
	aa3	/MUSIC (*) 4	(*) 1	Yes	2.80 or later
	m4a	/PSP/MUSIC	(*) 1	Yes	2.70~2.71
	m4a	/MUSIC (*) 4	(*) 1	Yes	2.80 or later
	3gp	/MUSIC (*) 4	(*) 1	Yes	2.80 or later
video/vnd.sony.mp4-m4v	-	/MP_ROOT/???MNV01	(*) 2	No	2.00 or later
video/vnd.sony.mp4-maq	-	/MP_ROOT/???ANV01	(*) 2	No	2.00 or later
video/vnd.sony.mp4-mav	-	/MP_ROOT/???ANV01	(*) 2	No	3.00 or later
video/vnd.sony.mp4-mas	-	/MP_ROOT/???ANV01	(*) 2	No	3.00 or later
video/vnd.sony.mp4-mah	-	/MP_ROOT/???ANV01	(*) 2	No	3.00 or later
video/*	-	/VIDEO	(*) 1	Yes	2.80 or later
x-video/*	-	/VIDEO	(*) 1	Yes	2.80 or later

MIME-Type	Extension	Folder	File Name	Can be Changed	Supporting System
-	mp4	/VIDEO	(*) 1	Yes	2.80 or later
-	m4v	/VIDEO	(*) 1	Yes	2.80 or later
-	avi	/VIDEO	(*) 1	Yes	3.00 or later
application/x-psp-theme	-	/PSP/THEME	(*) 1	No	3.70 or later
-	ptf	/PSP/THEME	(*) 1	No	3.70 or later
application/x-ssj-peta-guide	-	/MSSSJ/GUIDE	(*) 1	No	3.70 or later
-	sbk	/MSSSJ/GUIDE	(*) 1	No	3.70 or later
-	sb2	/MSSSJ/GUIDE	(*) 1	No	3.70 or later
application/x-ssj-peta-pemap	-	/MSSSJ/PE	(*) 1	No	3.70 or later
application/x-psp-radio-skin	-	/PSP/RADIOPLAYER	(*) 1	No	3.80 or later
-	prs	/PSP/RADIOPLAYER	(*) 1	No	3.80 or later
-	opml	/PSP/RSSCH/IMPORT	(*) 1	No	3.80 or later
Combination other than the above		/PSP/COMMON	(*) 1	Yes	-

(*) 1 File name conforms to server specification.

(*) 2 File name is determined by the Internet browser.

(*) 3 When the Internet browser is started up from an application on a UMD™ or on a Memory Stick™, the download destination may be /PSP/PHOTO.

(*) 4 When the Internet browser is started up from an application on a UMD™ or on a Memory Stick™, the download destination may be /PSP/MUSIC.

In most cases, the name of the download destination folder and the file name can be modified by the user. Although the file name can be edited arbitrarily by using the on-screen keyboard, the folder can only be selected from the download-target folders. Modification may also be restricted by a specification of the application that called the Internet browser.

(4) Download target folders

The download target folders are as follows.

Folder	Supporting System
/PICTURE	2.80 or later
/MUSIC	2.80 or later
/VIDEO	2.80 or later
/PSP/COMMON	2.00 or later
/PSP/PHOTO	2.00 or later
/PSP/MUSIC	2.00 or later
/MP_ROOT/???MNV01	2.00 or later
/MP_ROOT/???ANV01	2.00 or later

Uploads

The Internet browser supports the uploading of a file from a form. Although there are no specific constraints when directly entering a file name in the input box of an input tag, when the select button is pressed in the UI to select the file to be uploaded, only files in an uploadable folder can be selected.

(1) Uploadable folders

The uploadable folders are as follows.

Folder	Supporting System
/DCIM	2.70 or later
/PICTURE	2.80 or later
/PSP/PHOTO	2.00 or later
/PSP/SCREENSHOT	3.80 or later
/VIDEO/CHOTTO_SHOT	3.30 or later
/VIDEO/GO_EDIT	3.30 or later
/MSSSJ/GUIDE	3.70 or later
/PSP/COMMON	2.00 or later

Automatic HTTP Authentication

The Internet browser supports a function that automatically performs HTTP authentication without user intervention, by using authentication information specified from an application such as a game program.

(1) Application Conditions

Automatic HTTP authentication is applicable when the following conditions are satisfied.

- The Realm (AuthName) specification in the content begins with the Realm specified by the application
- The URL in the content begins with the URL specified by the application
  (*) Authentication information that contains the longest matching URL is used.
- The authentication method in the content matches the authentication method specified by the application

Only if all of these conditions are satisfied will HTTP authentication be automatically performed and the target page displayed.

(2) Server Settings

Server-side settings must enable HTTP authentication for the content. In addition, the following actions must be taken for all content for which automatic HTTP authentication is to be performed.

- Set the authentication method so that it matches what is used by the application
  (*) Only BASIC authentication or DIGEST authentication can be used.

- Set the Realm (AuthName) so that it is the same as the beginning of the string specified by the application
  (*) If no string is specified, "Auto-HTTP-Auth/1.0" is used as the default value.
- Set the user name and password so that they match those of the application
- Specify the URL from the application

For information about how to make these server settings, refer to the manual of the server software in use, or contact the server administrator.

4 Client Identification Information

The PSP™ Internet browser provides a means of obtaining information that can be used effectively for client identification based on the standards that were described in the previous chapter. This kind of information can be used in a server program, CGI, or JavaScript to perform dynamic content distribution, process switching, and browsing control.
This chapter describes the various kinds of information that can be used for client identification.

Request Header

Identification information that can be referenced from a server program or from a CGI as server variables is sent from the Internet browser in the HTTP request header.
The request header contains the following information related to client identification.

(1) User-Agent

Header Content

User-Agent: Mozilla/4.0 (PSP (PlayStation Portable); 2.00)

Description

This is a standard header representing the Internet browser type and version.
A CGI references it by using the variable HTTP_USER_AGENT.
Although there are no plans for changing the content of this header when the system software version is upgraded, it may be changed in a future version if functions are significantly changed or extended and it becomes necessary for sites to support those functions.

Supporting System

2.00 or later

(2) Accept-Language

Header Content

Accept-Language: xx-xx

Description

This is a standard header representing the language that is permitted by the Internet browser (user).
A CGI references it by using the variable HTTP_ACCEPT_LANGUAGE.
The Internet browser determines the languages that are permitted according to the

console's display language setting. The content references this header as necessary to switch the language that is to be displayed.

The xx-xx part is defined as follows.

Display Language Setting	xx-xx	Supporting system
German	“de”	2.00 or later
English	“en”	2.00 or later
Spanish	“es”	2.00 or later
French	“fr”	2.00 or later
Italian	“it”	2.00 or later
Dutch	“nl”	2.00 or later
Portuguese	“pt”	2.00 or later
Russian	“ru”	2.00 or later
Japanese	“ja”	2.00 or later
Korean	“ko”	2.00 or later
Simplified Chinese	“zh-cn”	2.70 or later
Traditional Chinese	“zh-tw”	2.70 or later

Example

```
Accept-Language: de
```

(3) x-psp-productcode

Header Content

x-psp-productcode: XXX

Description

This is a PSP™ specific extended header representing PSP™ console destination information.

A CGI references it by using the variable HTTP_X_PSP_PRODUCTCODE.

The content of this header will always be the same on the same PSP™ regardless of the user’s behavior.

Content may reference this header as necessary to guide the user to an appropriate page.

The XXX part is defined as follows.

Destination	XXX
Development tool	“TOOL”
Japan	“J1”
North America	“UC2”
Eastern Europe/European countries	“CEL”
Korea	“KR2”
UK	“CEK”
Mexico	“MX2”
AU/NZ	“AU3”

Destination	XXX
South Asia	“E12”
Taiwan	“TW1”
Russia	“RU3”
China	“CN9”

Example

```
x-psp-productcode: J1
```

Supporting System

2.50 or later

(4) x-psp-browser

Header Content

x-psp-browser: n.nn (xxx; yyy; zzz; …)

Description

This is a PSP™-specific extended header representing the Internet browser's system software version and start-up mode.

A CGI references it by using the variable HTTP_X_PSP_BROWSER.

n.nn represents the Internet browser's system software version. Although this basically is synchronized with the PSP™ console's system software version, a change in this version number may be deferred when a minor version upgrade is performed that does not affect the Internet browser's specifications.

Keywords representing start-up modes and other properties are entered for xxx, yyy, …

The order in which the keywords and properties appear is not defined.

The currently defined keywords and properties are as follows.

Keyword	Meaning	Supporting System
LX	Started up from the XMB™.	2.50 or later
SX	Started up from an application on the XMB™.	2.50 or later
LU	Started up from an application on a UMD™ or on a Memory Stick™.	2.50 or later

Property	Value	Supporting System
system	System software version	2.80 or later (LX,SX)
		3.00 or later (LU)

Example

```
x-psp-browser: 5.00 (LU; system=5.00)
```

Supporting System

2.50 or later

(5) x-psp-application

Header Content

x-psp-application: xxx…

Description

This is a PSP™ specific extended header representing the name or version of the application which launches the Internet browser.

A CGI references it by using the variable HTTP_X_PSP_APPLICATION.

xxx… is the string that the application specified. The Internet browser never changes it.

Example

```
x-psp-application: Everybodys Golf Portable/1.0.0 (UCJS-10001)
```

Supporting System

2.60 or later

JavaScript

Identification information that can be referenced from JavaScript code within content is built into the JavaScript engine of the Internet browser. Although there are no plans for changing this information when the system software version is upgraded, it may be changed if functions are significantly changed or extended in a future version and it becomes necessary for sites to support those functions.

The main identification information is shown below.

(1) navigator object

Property	Value	Supporting System
userAgent	"Mozilla/4.0 (PSP (PlayStation Portable); 2.00)"	2.00 or later
appCodeName	"Mozilla"	2.00 or later
appName	"PSP (PlayStation Portable) Internet Browser"	2.00 or later
appVersion	"2.00"	2.00 or later
platform	"PSP"	2.00 or later
language	"xx-xx" See table below for value of xx-xx.	3.70 or later

System Language Setting	xx-xx
German	"de"
English	"en"
Spanish	"es"
French	"fr"

System Language Setting	xx-xx
Italian	"it"
Dutch	"nl"
Portuguese	"pt"
Russian	"ru"
Japanese	"ja"
Korean	"ko"
Simplified Chinese	"zh-cn"
Traditional Chinese	"zh-tw"

(2) screen object

Property	Value
availWidth	480
availHeight	272
width	480
height	272
colorDepth	32
pixelDepth	32

In addition to the above, there is also identification information that can be acquired from JavaScript code using a PSP™ extended plug-in. For details, see the section entitled "PSP™ extended plug-in".

5 Flash® Player Plug-in

The PSP™ Internet browser supports playback of Flash® content using a Flash® Player plug-in.
This section presents an overview of Flash® Player functions that are built in to PSP™ system software 5.00.

Version

The version of the Flash® Player for the PSP™ system software ver 5.00 is 6 (6,0,72,27).
The basic features conform to the specifications of the Flash® Player with the version described above.

Supported Devices

The Flash® Player for PSP™ system software ver 5.00 supports the following input and output devices.

(1) Mouse

Analog stick and the Enter button work as one-button mouse.
Pointer movement, click and drag & drop features are supported.

(2) Keyboard

The directional keys work as the up, down, left, right cursor keys of a keyboard.

(3) Character input

Supports character input in text boxes using the on-screen keyboard.

(4) Fonts

Supports device fonts.
The applicable scope of the font sizes and styles are the same as the Internet Browser.

(5) Sound

Supports PCM, ADPCM and MP3 audio/sound playback for the speaker and headphone.

Unsupported Functions

The following features are not supported for the Flash® Player for PSP™ system software ver 5.00.

(1) Clipboard

Text copy, cut and paste features are not supported.

(2) Video

The playback of video data such as H.263, Sorenson Video, and Motion JPEG is not supported.

(3) Context menu

Context menu display, control or the features that are normally included in them are not supported.

(4) Printing

Printing is not supported.

(5) Live Connect

The communication feature between JavaScript and Flash® Player plug-ins is not supported.

(6) FCS (Flash Communication Server)

A connection to FCS, or features that requires it are not supported.

Related functions

- HTTP Tunneling
- Screen Sharing

(7) XMLSocket

Continuous connection and communication with the server using the XMLSocket feature is not supported.

(8) Transparent background display

Displaying the background in transparent by specifying the wmode to “transparent” is not supported.

(9) Streaming

Streaming playback of sound, etc. is not supported.

Known problems and restrictions

The Flash® Player for PSP™ system software ver 5.00 has the following limitations.

(1) Size of content

Large-size Flash® content cannot be played.

Although it depends on the content, on average, the amount of memory required by the Flash® Player is approximately three times the size of the Flash® content file.

It follows that if the Flash® content is placed in the HTML, then the size of the content in a page should be within 1.5 MB approximately.

(2) Loading external files

If a function such as loadVariable() is used to load data from an external file, the timing when data is loaded may be different from that of PCs.

Therefore, the completion of loading must be confirmed.

(3) Size of device fonts

The device font has five sizes, just like the Internet browser.

It may not be possible to achieve the same layout as on a PC because content may not fit as expected.

(4) Generating mouseout events

A mouseout event may not occur when the pointer gets out of the content if objects are arranged on the edge of the content field.

Exercise caution in these cases by anticipating when content will be displayed in unexpected ways.

(5) Nesting of ActionScript

Restrictions when deep nesting occurs with ActionScript are different from PCs. Nesting is terminated at a shallower level compared to a PC.

6 PSP™ Plug-in

The PSP™ Internet browser provides functions for cooperatively linking together content with an application such as a game program. This is done by defining a dedicated PSP™ plug-in and writing JavaScript code so that the plug-in can be used by the content to support the application.
This section explains how to use these functions provided by PSP™ plug-ins, from the content..

Supported Functions

The following application linking functions are currently supported by PSP™ plug-ins.

Linking Function	Supporting System
Writing a return value to an application	2.70 or later
Shutting down the Internet browser	2.70 or later

Note that a specification is required by the application to use the PSP™ plug-in linking functions, and, furthermore, these functions are disabled in an Internet browser that is started up from the XMB™.

Plug-in Definition

To use the linking functions, provide an object definition for the PSP™ plug-in in the HTML as follows.

Example

```
<object name="psp" type="application/x-psp-plugin"></object>
```

Although the name attribute is set to "psp" in the example above, an arbitrary name can be specified. The object is handled in the JavaScript by using the name specified here.

Writing a Return Value to the Application

A return value is written to the application by having the JavaScript in the content write data to an application buffer.
The following example shows how to write a return value from the content.

Example

```
<script language="JavaScript">
<!--
```

```
a = 10;
b = new Array(3);
b[0] = 1; b[1] = 2; b[2] = 3;
if (psp) {
    psp.save(a);
    psp.save(b);
}
//-->
</script>
```

The save() method is used to write data. The data that is written will be a string with a format based on a JavaScript internal representation. For example, in the example above, if “a” is used as the return value, "10" is written to the buffer, and if “b” is used as the return value, "1,2,3" is written to the buffer.
Also, if data is written multiple times while the Internet browser is active, the buffer is always overwritten by the data that was written last. In the example above, the b value will be valid.

Shutting Down the Internet Browser

The Internet browser can be shut down according to a specification in the content. This function can only be implemented in the content if the PSP™ plug-in is enabled.
The following example shows how to shut down the Internet browser from the content.

Example

```
<script language="JavaScript">
<!--
a = 10;
if (psp) {
    psp.save(a);
    psp.exit();
}
//-->
</script>
```

The exit() method is used to shut down the Internet browser. In the example above, "10" is written by the save() method as the return value and then the browser is shut down by the exit() method.

When the exit() method is executed, the behavior of the Internet browser is the same as if a shutdown operation were performed by the user.

Sample

The sample code shown below includes a plug-in validity check.

Sample Code

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
<title>Sample for the PSP plugin</title>
</head>
<body>
<script type="text/javascript">
function exit_browser()
{
    if (psp) {
        psp.exit();
        return true;
    }
    return false;
}

function write_result1()
{
    a = 10;
    if (psp) {
        psp.save(a);
        return true;
    }
    return false;
}

function write_result2()
{
    a = 10;
    b = new Array(3);
    b[0] = 1; b[1] = 2;  b[2] = 3;
    if (psp) {
        psp.save(a);
        psp.save(b);
        return true;
    }
    return false;
}
</script>
<p>
<button onclick="write_result1();">click to write result (1)</button>
required:"10"<br/>
<button onclick="write_result2();">click to write result (2)</button>
```

```
required:"1,2,3"<br/>
<button onclick="exit_browser();">click to exit</button><br/>
</p>
</body>
<script type="text/javascript">
if (navigator.mimeTypes && navigator.mimeTypes["application/x-psp-plugin"]) {
    var plugin = navigator.mimeTypes["application/x-psp-plugin"].enabledPlugin;
    if (plugin) {
        document.write("<object name=\"psp\"
type=\"application/x-psp-plugin\"></object>\n");
    }
};
//
// below is also OK.
//
//if (navigator.plugins) {
//    var plugin = navigator.plugins["psp plugin"];
//    if (plugin) {
//        document.write("<object name=\"psp\"
type=\"application/x-psp-plugin\"></object>\n");
//    }
//}
</script>
</html>
```

The plug-in validity check is performed by using MIME-Type "application/x-psp-plugin."
The object tag is described below the code enclosed by <body> and </body>. This is because in the current system software version, blanks are rendered at the position where the object tag is described. The object tag should be described at a similar position as in the sample code so that there is no effect when the trouble is corrected in a future system software version.

7 PSP™ Extended Plug-in

In addition to application linking functions provided by PSP™ plug-ins, the PSP™ Internet browser also supports linking functions provided by valid PSP™ extended plug-ins even when the Internet browser is started up from the XMB™
This section explains how to use functions provided by PSP™ extended plug-ins, from the content.

Supported Functions

The following application linking functions are currently supported by PSP™ extended plug-ins.

Linking Function	Supporting System
Getting Internet browser extended header information	2.80 or later
Getting maximum amount of content heap used	2.80 or later
Getting total amount of content heap	3.70 or later

In addition to the above, other functions may also be available that have been registered by applications, such as game programs.

Plug-in Definition

To use the linking functions, include an object definition for the PSP™ extended plug-in in the HTML content as follows.

Description Example

```
<object name="pspext" type="application/x-psp-extplugin"></object>
```

Although the name attribute is set to "pspext" in the example above, an arbitrary name can be specified. In the JavaScript code, the object is handled by using the name specified here.

Getting Internet Browser Extended Header Information

The Internet browser can get the contents of an extended header that is appended when a request is sent to the server. This allows the client to be identified more precisely in the JavaScript code.
To get extended header information, specify the following.

Description Example

```
<script language="JavaScript">
<!--
if (pspext) {
    psp_browser = pspext.sysGetEnv('x-psp-browser');
}
//-->
</script>
```

The sysGetEnv() method is used to get extended header information. You can get the contents of a header by calling this method and specifying the desired header name.
The following header information can be acquired.

Header	Supporting System
x-psp-productcode	2.80 or later
x-psp-browser	2.80 or later
x-psp-application	2.80 or later
User-Agent (*)	2.80 or later

(*) Although User-Agent is a standard header, it can also be acquired.

Getting the Maximum Content Heap Size Used

You can get the peak value of the amount of heap memory that was used by the content. This facilitates the production of content by allowing you to find out the approximate amount of heap memory necessary to display the content.
Note that the peak value is the total amount of heap memory used by all tabs, and includes the standard heap memory that is used by the system.
To get the peak value of the amount of heap memory used, specify the following.

Description Example

```
<script language="JavaScript">
<!--
if (pspext) {
    peak = pspext.sysGetHeapUsePeak();
}
//-->
</script>
```

The sysGetHeapUsePeak() method is used to get the peak value of the amount of heap memory used. The peak value is acquired as an integer in units of bytes.

Getting the Total Content Heap Size

You can get the total amount of heap memory allocated for content by the system or by an application such as a game program. You can use this information to change the behavior of the content as appropriate.
Note that the content heap is shared by all tabs and is also used by the system.
To get the total size of content heap memory used, specify the following.

Description Example

```
<script language="JavaScript">
<!--
if (pspext) {
    size = pspext.sysGetTotalHeapSize();
}
//-->
</script>
```

The sysGetTotalHeapSize() method is used to get the total size of the content heap. This method returns the total size as an integer in units of bytes.

Sample

The following sample code also includes a plug-in validity check.

Sample code

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
<title>Sample for the PSP ext-plugin</title>
</head>
<body>
<script type="text/javascript">
function get_system_version()
{
    system_version = null;
    if (pspext) {
        psp_browser = pspext.sysGetEnv('x-psp-browser');
    }
    if (psp_browser) {
        start = psp_browser.indexOf('system=', 0) + 'system='.length;
        end   = psp_browser.indexOf(';', start);
        if (end == -1) {
            end   = psp_browser.indexOf(')', start);
        }
        system_version = psp_browser.substr(start, end-start);
```

```
    }
//    return system_version;
    alert('system_version='+system_version);
}

function get_heap_peak()
{
    heap_peak = -1;
    if (pspext) {
        heap_peak = pspext.sysGetHeapUsePeak();
    }
//    return heap_peak;
    alert('heap_peak='+heap_peak);
}

function get_heap_size()
{
    heap_size = -1;
    if (pspext) {
        heap_size = pspext.sysGetTotalHeapSize();
    }
//    return heap_size;
    alert('heap_size='+heap_size);
}
</script>
<p>
<button onclick="get_system_version();">click to get system version</button>
required:"5.00"<br/>
<button onclick="get_heap_peak();">click to get heap use peak</button> required:some
numeric value<br/>
<button onclick="get_heap_size();">click to get total heap size</button>
required:some numeric value<br/>
</p>
</body>
<script type="text/javascript">
if (navigator.mimeTypes && navigator.mimeTypes["application/x-psp-extplugin"]) {
    var plugin = navigator.mimeTypes["application/x-psp-extplugin"].enabledPlugin;
    if (plugin) {
        document.write("<object name=\"pspext\"
type=\"application/x-psp-extplugin\"></object>\n");
    }
};
//
// below is also OK.
//
//if (navigator.plugins) {
//    var plugin = navigator.plugins["psp extplugin"];
//    if (plugin) {
//        document.write("<object name=\"pspext\"
type=\"application/x-psp-extplugin\"></object>\n");
//    }
```

```
//}
</script>
</html>
```

The plug-in validity check is performed by MIME-Type "application/x-psp-extplugin."
The object tag is specified below the code enclosed by <body> and </body>. This is done so that with the current version of system software, blanks are rendered at the position where the object tag is specified. The object tag should be specified at a similar position as in the sample code so that there is no effect when the trouble is corrected in a future version of system software.

8 Appendix

Detailed Specifications for HTML

The Internet browser supports HTML 4.1, XHTML 1.1, and XHTML Basic. A list of elements that are supported by the browser appears below. However, note that attributes that appear under Unsupported Attributes cannot be used even if they are entered for Available Elements. Also, the following attributes are not supported for any element.

- ondblclick
- xml:lang
- xmlns

Available Elements	Unsupported Attribute	Available Elements	Unsupported Attribute	Available Elements	Unsupported Attribute
<HTML>	fragments	<BDO>	lang	<BLOCKQUOTE>	
<HEAD>	profile	<EM>		<Q>	
<TITLE>		<STRONG>		<SUB>	
<META>	scheme	<DFN>		<SUP>	
<BODY>	nowrap	<CODE>		<P>	
<DIV>	nowrap	<SAMP>		 	
<SPAN>		<KBD>		<PRE>	width, cols, wrap
<H1>		<VAR>			
<H2>		<CITE>		<INS>	datetime
<H3>		<ABBR>		<DEL>	datetime
<H4>		<ACRONYM>		<PLAINTEXT>	
<H5>				<XMP>	
<H6>				<BLINK>	
<ADDRESS>				<MARQUEE>	truespeed
<UL>	compact	<COLGROUP>	align, char, charoff	<A>	hreflang, type, rel, rev, charset
<OL>	compact, segnum				
		<COL>	align, char, charoff	<LINK>	Style, onclick, onmousedown, onmous
<LI>	compact				
<DL>	compact	<TR>	char, charoff, width,		
<DT>					
<DD>					

Available Elements	Unsupported Attribute	Available Elements	Unsupported Attribute	Available Elements	Unsupported Attribute
<DIR>	compact		bordercolordark, bordercolorlight,		eup, onmouseover, onmousemove, onmouseout, onkeypress, onkeydown, onkeyup, hreflang, rev, target, charset
<MENU>	compact	<TH>	headers, scope, abbr, axis, char, charoff, bordercolordark, bordercolorlight		
<TABLE>	summary, bordercolordark, bordercolorlight, cols, hspace, vspace			<BASE>	
<CAPTION>	valign	<TD>	headers, scope, abbr, axis, char, charoff, bordercolordark, bordercolorlight		
<THEAD>	align, char, charoff, valign				
<TFOOT>	align, char, charoff, valign				
<TBODY>	char, charoff				

<IMG>	longdesc	<EMBAD>	align, alt, border, code, codebase, frameborder, name, palette, pluginspage, pluginurl, units, optional_parm	<FONT>	face
<IMAGE>				<BASEFONT>	face
<OBJECT>	classid, codebase, codetype, archive, standby, tabindex, align, border, accesskey, code			<HR>	
				<FORM>	
				<INPUT>	
				<BUTTON>	
		<NOEMBED>		<SELECT>	
		<MAP>		<OPTGROUP>	style
<PARAM>	valuetype	<AREA>	alt	<OPTION>	
<APPLET>	codebase, code, name, archive, object, width, height, id, class, title, style, alt, align, hspave,	<STYLE>	xml:space	<TEXTAREA>	onselect
		<CENTER>		<LABEL>	onfocus, onblur
		<TT>		<FIELDSET>	align
		<I>		<SCRIPT>	
		<B>		<FRAMESET>	framespacing
		<BIG>		<FRAME>	longdesc
		<SMALL>		<NOFRAMES>	
		<STRIKE>		<IFRAME>	

Available Elements	Unsupported Attribute	Available Elements	Unsupported Attribute	Available Elements	Unsupported Attribute
	vspacve, mayscript, src	<S> <U>			

Detailed Specifications for CSS

The Internet browser supports CSS1 and part of CSS2. Unsupported properties are listed below.

(1) Unsupported properties

Properties	Properties	Properties	Properties	Properties
E:lang(cc) E:first-letter E:first-line E:before E:after	content quotes counter-reset counter-increment marker-offset	font-stretch font-size-adjust	volume speak pause-before pause-after pause cue-before cue-after cue play-during azimuth elevation speech-rate voice-family	pitch pitch-range stress richness speak-punctuation speak-numeral
unicode-bidl	size marks page-break-before page-break-after page-break-inside page orphans windows	text-shadow letter-spacing word-spacing text-transform		
min-width max-width height min-height max-height		table-layout speak-header		-wap-accesskey
		cursor outline outline-width outline-style outline-color		

(2) Properties with unsupported values

Properties	Values	Properties	Values
display	list-item marker run-in compact table inline-table table-row-group table-column table-column-group table-header-group table-footer-group table-row table-cell table-caption	list-style-type	hebrew armenian georgian cjk-ideographic hiragana katakana hiragana-iroha katakana-iroha
		font	icon menu message-box small-caption status-bar
position	Relative	text-align	justify <string>

Properties	Values	Properties	Values
overflow	Scroll auto		inherit
clip	<shape> inherit	caption-side	left right
visibilit y	Collapse	border-spacing	inherit
		empty-cells	show inherit

Detailed Specifications for JavaScript

The Internet browser supports part of Java Script 1.5 and ECMAScript 3rd Edition.

Unsupported objects are listed below.

The following method is not supported by any object.

- toSource()

Properties	Method	Statements
Objects	eval(string) unwatch(prop) watch(prop, handler)	export Statements import Statements
Function	arity	
Arguments	caller	
RegExp	$1, ... $9 input(also $_) lastMatch(also $&) lastParen(also $+) leftContext(also $`) rightContext(also $') multiline(also $*) toString() compile(pattern[, flags])	
Error	MemoryError	

Detailed Specifications for DOM

(1) DOM Level0

Object	Attribute	Method	Event Handler
Navigator	cookieEnabled oscpu vendor vendorSub	javaEnabled() preferens(in DOMString prefName[, in DOMString setValue]) savePreferences() taintEnabled()	
MimeType Array	.name		
Screen	availLeft availTop		

Object	Attribute	Method	Event Handler
Window	crypto defaultStatus locationbar menubar offscreenBuffering outerHeight outerWidth pageXOffset pageYOffset personalbar screen screenX screenY scrollbars status statusbar toolbar	atob(encodedData) blur() btona(stringToEncode) captureEvents(eventType1[,eventTypeN...]) disableExpternalCapture() enableExternalCapture() find(in DOMString string, in boolean caseSensitive, in boolean backward) focus() handleEvent(event) mobyBy(in long horizontal, in long vertical) mobeTo(in long x-coordnate, in long y-coordinate) print() releaseEvents(eventType1[, eventTypeN...]) resizeBy(in long horizontal, in long vertical) resizeTo(in long outerWidth, in long outerHeight) routeEvent(event) setHotKeys(in boolean setting) setResizable(in Boolean setting) setZOptions(windowPosition)	onabort onclose ondragdrop onerror onkeyup onmove onpaint onreset onresize onscroll onselect onsubmit new window()
History	current next previous		
Bar	All		
Crypto	All		
Layer (layers)	All		

(2) DOM Core Level 1, 2

Object	Attribute	Method
DOMException	code	
DOMImplementation		
DocumentFragment		

Object	Attribute	Method
Document	doctype	createAttribute(in DOMString name) createAttributeNS(in DOMString namespaceURI, in DOMString qualifiedName) createCDATASection(in DOMString data) createComment(in DOMString data) createDocumentFragment() createElement(in DOMString tagName) createElementNS(in DOMString namespaceURI, in DOMString qualifiedName) createEntityReferense(in DOMString name) createProcessinginstruction(in DOMString target, in DOMString data) createTextode(in DOMString data) getElementsByTagNameNS (in DOMString namespaceURI, in DOMString localName) importNode(in Node ImporteNode, in boolean deep)
Node	localName nameSpaceURI prefix	appendChild(in Node newChild) doneNode(in Boolean deep) hasAttributes() insertBefore(in Node newChild, in Node refChild) isSupported(in DOMString feature, in DOMString version) normalize() replaceChild(in Node newChild, in Node oldChild) removeChild(in Node oldChild)
NamedNodeMap		getNamedItemNS(in DOMString namespaceURI, in DOMString localName) removeNamedItem(in DOMString name) removeNamedItemNS(in DOMString namespaceURI, in DOMString localName) setNamedItem(in Node arg) setNamedItemNS(in Node arg)
Character Data		appendData(in DOMString arg) deleteData (in unsigned long offset, in unsigned long count) insertData (in unsigned long offset, in DOMString arg) replaceData(in unsigned long offset, in unsigned long count, in DOMString arg)
Attr	OwnerElement	
Element		getAttributeNodeNS(namespaceURI,localName) getAttributeNS(namespaceURI, localName); getElementsByTagNameNS(namespaceURI,localName) hasAttribute(name) hasAttributeNS(namespaceURI,localName) normalize() removeAttribute(in DOMString name) removeAttributeNode(in Attr oldAttr) removeAttributeNS(namespaceURI,localName) setAttributeNode(in Attr newAttr) setAttributeNodeNS(newAttr) setAttributeNS(namespaceURI,qualifiedName)
Text(TextNode)	All	

Object	Attribute	Method
Comment	All	
CDATASection	All	
DocumentType	All	
Notation	All	
Entity	All	
EntityReference	All	
Processinginstruction	All	

(3) DOM Event Level 2

Object	Attribute	Method	Event Handler
EventTarget		addEventListener() removeEventListener()	
Event	data h eigh layerX layerY modifiers pageX pageY		
EventException			
UIEventEvent	detail	initUIEvent()	
MouseEvent:UIEvent	altKey button ctrlKey metaKey relatedTarget shiftKey	initMouseEvent()	

(4) DOM HTML Level 1, 2

The following Event Handler is not supported by any object.

- Ondblclick

Object	Attribute	Method	Event Handler
HTMLDOMImplementation	All		
HTMLDocument	formName height ids layers tags	captureEvent(event Type) getSelection() handleEvent(event) releaseEvents(eventType) routeEvent(event)	

Object	Attribute	Method	Event Handler
HTMLLinkElement			onclick onkeydown onkeyup onmousedown onmouseout onmouseover onmouseup
HTMLFormElement	encoding	handleEvent(event)	onclick onkeydown onkeypress onkeyup onmousedown onmousemove onmouseout onmouseover onmouseup onreset onsubmit
HTMLSelectElement		handleEvent(event)	
HTMLOptionElement		newOption([text[, value[, defaultSelected [, selected]]]])	
HTMLInputElement	length	handleEvent(event) select()	onselect
HTMLLabelElement			onblur onfocus
HTMLAnchourElement	text x y		
HTMLImageElement		handleEvent(event)	onabort onerror
HTMLObjectElement	contentDocument		onclick onkeydown onkeypress onkeyup onmousedown onmousemove onmouseover onmouseup
HTMLAppletElement	All		
HTMLScriptElement	event htmlFor		
HTMLTableElement		createCaption() createTFoot() createTHead() deleteCaption() deleteTFoot() deleteTHead() deleteRow(in long index) insertRow(in long index)	

Object	Attribute	Method	Event Handler
HTMLTableSectionElement		deleteRow(in long index) insertRow(in long index)	
HTMLTableRowElement		deleteCell(in long index) insertCell(in long index)	
HTMLFrameElement	contentDocument		
HTMLIFrameElement	contentDocument		

(5) DOM Style Level2、Document Object Model CSS

Object	Attribute	Method	Event Handler
CSS2Properties	azimuth borderCollapse borderSpacing bottom captionSide clip content counterIncrement counterReset cue cueAfter cueBefore cursor elevation emptyCells fontSizeAdjust fontStretch fontVariant letterSpacving markerOffset marks maxHeight maxWidth minHeight minWidth orphans outline outlineColor outlineStyle outlineWidth overflow page pageBreakAfter pageBreakBefore pageBreakInside pause pauseAfter		

Object	Attribute	Method	Event Handler
	pauseBefore pitch pithRange playDuring quotes richness right size speak speakHeader speakNumeral speakPunctuation speechRate stress tableLayout textShadow textTransform voiceFamily volume windows wordSpacing -wap-marquee -wap-marquee-style -wap-marquee-loop -wap-marquee-dir -wap-marquee-spee d -wap-accesskey -wap-input-format -wap-input-required		

Initial Values and Constraints

The following table shows initial values and constraints of items that can be changed by user operations, user settings, or specifications from applications.

Type	Item	Initial Value	Constraint/Range	Supporting System
HTTP	Connection timeout interval	60 seconds	(Fixed)	2.00 or later
	Send timeout interval	60 seconds	(Fixed)	2.00 or later
	Receive timeout interval	120 seconds	(Fixed)	2.00 or later

Type	Item	Initial Value	Constraint/Range	Supporting System
Cache	Cache size	512 KB	Off / 512 KB / 1024 KB / 2048 KB	2.00 or later
Cookie	Cookie reception	Receive	Receive / Do not receive / Verify	2.00 or later
	Size	-	Up to 4 KB	2.00 or later
	Number	-	Up to 20 per domain	2.00 or later
Display	Image display	On	On / Off	2.00 or later
	Animation display	On	On / Off	2.00 or later
	JavaScript	On	On / Off	2.00 or later
	Character size	Standard	Small/ Standard / Large	2.00 or later
	Conserve Memory	Off	On/Off	3.10 or later
	Display mode	Standard	Standard / Just fit / Smart fit	2.00 or later
Tab	Number of tabs	3	1 to 3	2.00 or later
Character input	Address input	-	Up to 512 characters	2.00 or later
	Home page setting	-	Up to 512 characters	2.00 or later
	Form input	-	Up to 512 characters	2.00 or later
Bookmark	Number saved	-	Up to 1000	2.00 or later
	Number of address characters	-	Up to 512 characters	2.00 or later
	Number of title characters	-	Up to 512 characters	2.00 or later
Browsing history	Number saved	-	Up to 100	2.00 or later
Input history	Address input history	-	Up to 100	2.00 or later
	Form input history	-	Up to 100	2.50 or later

9 Document History

Edition	Section	Subsection	Changes
1st	-	-	- Supported additional changes in system software version 5.00